VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

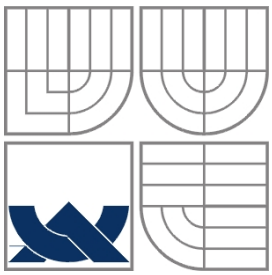
APLIKACE PRO SPRÁVU PROJEKTŮ DLE METODY GTD PRO WINDOWS MOBILE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

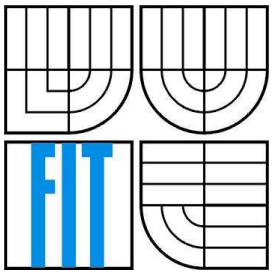
AUTOR PRÁCE
AUTHOR

LIBOR WEIGL

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

APLIKACE PRO SPRÁVU PROJEKTŮ DLE METODY GTD PRO WINDOWS MOBILE

GTD PROJECT MANAGEMENT FOR WINDOWS MOBILE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

LIBOR WEIGL

VEDOUCÍ PRÁCE
SUPERVISOR

ING. PETER SOLÁR

BRNO 2010

Abstrakt

Práce se zabývá metodou GTD a vývojem aplikace usnadňující její agendu. Snaží se analyzovat tuto metodu organizace času pro řízení projektů. Dále popisuje platformu Windows Mobile, pod kterou aplikace poběží. Též se zmíní o nástrojích použitých při vývoji této aplikace.

Abstract

This work deals with the GTD method and application, to facilitate its development agenda. The method allows us to become familiar with the organization of projects. It describes the Windows Mobile platform, under which applications will run, and outlines the tools used to develop this application.

Klíčová slova

Metoda GTD, Windows Mobile, .NET Compact Framework, Microsoft Visual Studio,CAB, Form, David Allen, Mít vše hotovo, Getting things Done, C#

Keywords

GTD, Getting Things Done, David Allen, CAB, Microsoft Visual Studio, .NET Compact Framework, C#

Citace

Weigl Libor: Aplikace pro správu projektů pro Windows Mobile, bakalářská práce, Brno, FIT VUT v Brně, 2010

Aplikace pro správu projektů dle metody GTD pro Windows Mobile

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing Petera Solára

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Libor Weigl
14. 5 2010

Poděkování

Rád bych tímto poděkoval vedoucímu mé práce Ing. Peteru Solárovi za odborné vedení, cenné rady a jeho čas, který mi věnoval během konzultací.

Obsah

1 Úvod

Dnešní doba klade na člověka vysoké požadavky. Ty se navíc neustále dynamicky mění. Při vystavení organizmu mimořádným podmínkám může dojit ke stavu, který je známy pod pojmem stres. Nejsme schopni se zbavit napětí, zátěže, tedy stresu úplně. To bohužel není možné, ale můžeme se to pokusit snížit na přijatelnou míru. Kdyby stres neodborně řečeno přerostl přes hlavu, mohli bychom skončit na psychiatrii, v horším případě na patologii. Je tedy důležité snažit se omezit stres v našem životě. K tomuto účelu nám muže posloužit metoda GTD. Tato bakalářská práce si klade za cíl nastudovat metodu GTD. Na základě ní vytvořit aplikaci pro Windows Mobile, která bude sloužit pro podporu GTD metody. Tato technická zpráva popisuje procesy, jež vedly k dosažení cíle bakalářské práce. Skládá se z devíti kapitol.

V druhé kapitole se seznámíme se základy metody Getting things Done. Ty uplatníme při návrhu aplikace GTD.

Třetí kapitola nám představí platformu operačního systému Windows Mobile a konkurenční platformy na dnešním trhu. Prozkoumáme Windows Mobile pod různými úhly pohledu. Neopomeneme zmínit základní programovou vybavenost. Poslední část kapitoly obsahuje popis nástrojů pro vývoj aplikací pod tímto operačním systémem. Poukazuje na jejich postupné zlepšování.

Ve čtvrté kapitole se dočteme o návrhu aplikace GTD. Především o rozdělení architektury do tři vrstev.

Pátá kapitola nám ukáže praktické využití vývojových nástrojů, se kterými jsme se seznámili ve druhé kapitole. Dále popisuje implementaci jednotlivých vrstev aplikace GTD. Na diagramech názorně uvidíme vrstvy aplikace reprezentované jednotlivými třídami.

V šesté kapitole bude vysvětleno ovládání a jeho možné způsoby. Dále se zde dozvíme minimální hardwarové nároky aplikace na zařízení.

Sedmá kapitola nás seznámí s testováním aplikace. Získáme představu o nástrojích, na kterých se aplikace testovala a jakým způsobem.

Osmá kapitola poukazuje na skrytý potenciál využití při řízení firemních procesů. Nastiňuje cestu, jakou by se měla tato práce dále ubírat. Ale i obavy z nové platformy.

Devátá poslední kapitola se věnuje souhrnu popsaných informací. Snaží se vytvořit celistvý obraz zadání a vývoje této bakalářské práce. Konfrontuje výsledky práce se zadáním.

2 Mít vše hotovo

2.1 Úvod

Mít vše hotovo, je převzatý název z anglického překladu Getting think done. Mezi uživateli se však ujal název, tvořený z prvních písmen anglického názvu, proto se již budeme dále v textu setkávat s touto zkratkou

GTD je metoda pro organizaci času. Autorem je David Allan, který své zkušenosti z dvacetileté praxe konzultanta pro zvyšování produktivity práce využil při vývoji metody GTD. Jeho pojetí organizace času přináší mnoho revolučních novinek. Zapojením této metody do našeho osobního života odbouráme stres, nebudeme zahlceni prací, budeme mít komplexní přehled o tom, co děláme a proč to děláme. Největším problémem při používání GTD budeme mi sami. Lidé se hůře přizpůsobují a tato metoda vyžaduje získání nových návyků a jejich dodržování. Jen na nás záleží, jak moc budeme úspěšní se začleněním GTD do našeho běžného života

Tato kapitola čerpá informace z knihy [1].

2.2 Metoda GTD

2.2.1 Proces: Řízení činností

Je možné tento proces natrénovat a tím dosáhnout lepších výsledků s vyšší mírou kontroly. Je důležité si řádně definovat cíle. Člověk si udělá představu, čeho by chtěl dosáhnout. Následný úkol pak nebude neřešitelný. Díky tomuto postupu získáme představu o čase potřebnému k jeho splnění. Pokud chceme mít tyto záležitosti náležitě pod kontrolou, měli bychom k nim přistupovat „zdola nahoru“, kdy se nejdříve zaměříme na nejmenší aktuální činnosti. Je také možné použít přístup „shora dolů. Problémem u tohoto řízení je, že nejsme schopni se zcela objektivně zaměřit na širší horizont. Pro kontrolování svých závazků a projektů je výhodné využívat „horizontální“ a „vertikální řízení“.

Pomocí horizontálního řízení udržujeme kontrolu nad všemi aktivitami, do kterých jsme zapojeni. Během dne můžeme zaznamenat jakoukoliv ze všech položek, které máme potřebu řešit, nebo se nám dožadují. Tento způsob řízení nám bude flexibilně přesouvat pozornost na právě aktuální položku.

Vertikální řízení se zaměřuje jen na konkrétní položku. Jedná se o projektové plánování. Hlavním účelem vertikálního a horizontálního řízení je dostat všechny informace z hlavy a vyplnit je. Dosáhneme pocitu uvolnění a nebudeme znát pojem „nemám čas“. Hlavní změnou oproti ostatním metodám organizace času bude, že se vše budeme snažit dostat z hlavy ven a následně zaznamenávat na jakékoliv paměťové medium. K záznamu můžeme využít například list papíru, specializovaný

software a hlasový záznamník. Pokud se nám podaří přesunout všechny úkoly, které udržujeme v mozku na papír, tak odpadne nepříjemný stav, kdy si je musíme neustále připomínat a na většinu bohužel zapomeneme. Poznámky uložené mimo naši hlavu se nám lepe utřídí. Naše mysl nebude skákat z jednoho nedokončeného úkolu na druhý, ale zaměří se jen na aktuální. Ten se nás bude snažit posunout blíže k vytouženému cíli.

2.2.2 Kontrola nad pracovním procesem

Základní metoda k dokonalému zvládání závazků bez stresu. Metoda se skládá z pěti oddělených fází. Závisí na nich naše úroveň organizace práce. Tato úroveň bude dána nejslaběji definovanou fází. A proto dodržování těchto fází je nesmírně důležitě. Pro jednotlivé fáze je dobré pravidlo si vyhranit určitou část dne. Například některým uživatelům vyhovuje ráno provést fázi sběru.

Seznam pěti fází:

1. Sběr
2. Zpracování
3. Uspořádání
4. Hodnocení
5. Provádění

2.2.2.1 Sběr

Při fázi sběru je nesmírně důležité, abychom dostali z hlavy všechny záležitosti, jimiž se náš mozek musí zabývat a tím se nám dostává pocit, že je nemáme pod kontrolou. Musíme vyloučit možnost vzniku děr, k tomu nám pomohou sběrná místa. Některá sběrná místa pracují nezávisle na nás. Téměř každý z nás již vlastní emailovou schránku, do které každý den přichází mnoho požadavků. Požadavek pro nás znamená jistý úkol, musíme se jím zabývat a snažit se ho dokončit. Podobně ukládáme požadavky, které se nám nahromadily v hlavě. Určíme si místo, kam je budeme ukládat. Do naší sběrné schránky uložíme všechny požadavky, které držíme v hlavě. Schránka může být například krabice, do které budeme dávat popsané listy papíru našimi úkoly. Počet těchto sběrných míst záleží jen na nás, čím méně, tím lépe. V určitých intervalech je nezbytné schránku vyprázdnit. Při procesu vyprazdňování se do schránky nesmí nic vracet, vše se musí zpracovat. Nedodržení tohoto pravidla vede k hromadění požadavků ve schránce, které nakonec nebudeme efektivně využívat.

2.2.2.2 Zpracování

Můžeme zpracovávat pouze ty požadavky, které byly nejprve uloženy do sběrné schránky a následně vybrány. Při rozhodování co uděláme s daným požadavkem, si musíme klast otázky. Na začátku si vždy položíme otázku: „Je to realizovatelné?“. Na to jsou správně jen dvě odpovědi „ano“ a „ne“. Další otázkou na danou záležitost je „nevyžaduje žádný další krok“. Pokud jsme odpověděli „ne“, máme na výběr z těchto tří možností.

1. Je to na vyhození, nebude to už potřeba.
2. Nyní není nutná žádná akce, ale možná se to bude moci zrealizovat v budoucnu (nechat uzrát, odložit)
3. Položka obsahuje potenciálně užitečnou informaci, která může být později potřebná (archivovat)

Pokud by naše odpověď byla na předchozí otázky „Ano“. Budeme si muset položit další sadu otázek.

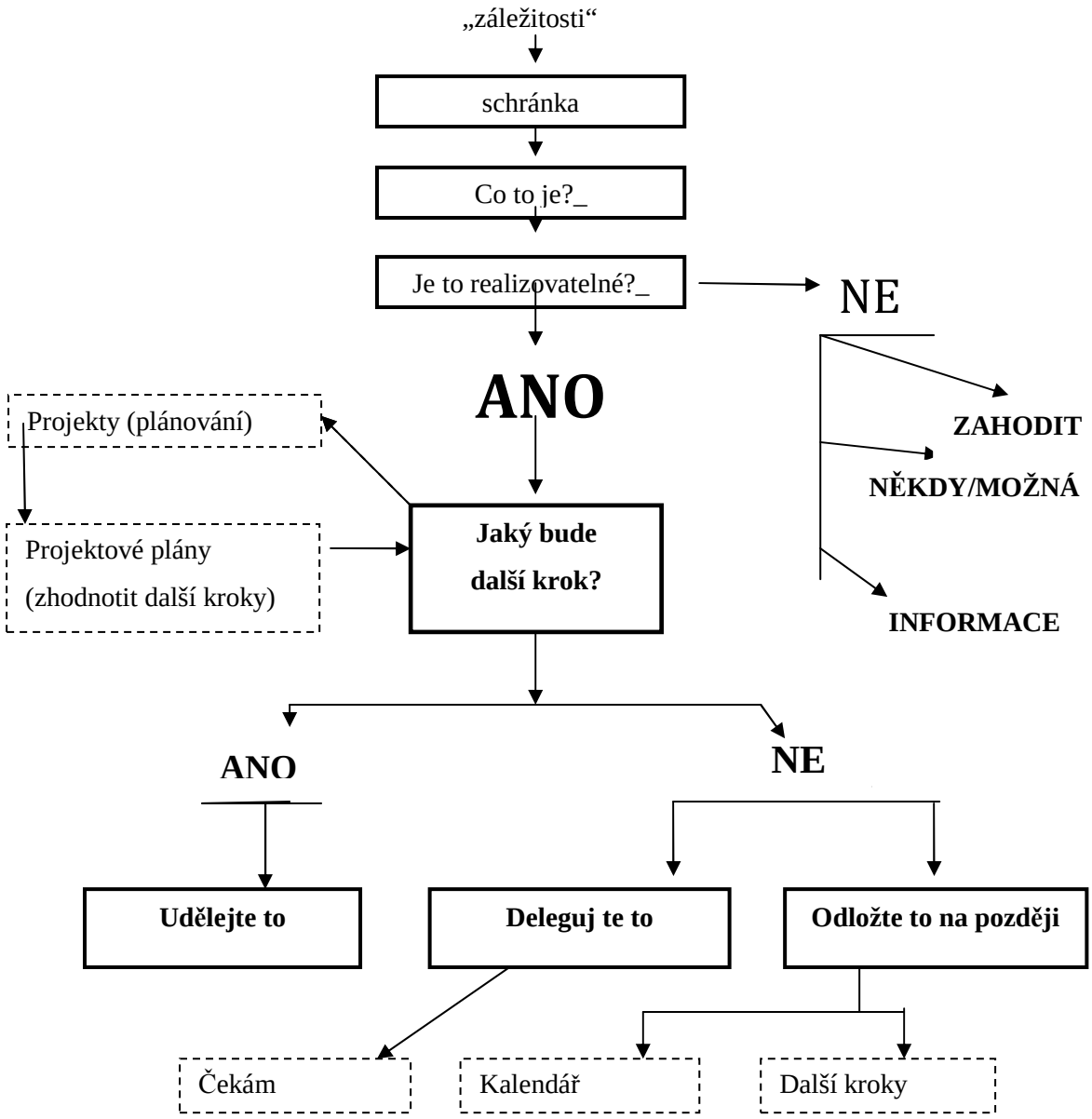
1. K jakému projektu jste se zavázali?
2. Jaký je další nutný krok?

Jestliže jste odpověděli, že jste se zavázali k nějakému projektu, poté si toto ustanovení uložte do seznamu projektů. Záznam v tomto seznamu slouží jako upozornění, že máte nevyřešený problém a pomůže vám při týdenním hodnocení vaší práce. Na otázku, jaký je nutný další krok, je dobré si nejdříve definovat tento pojem.

Další krok je příští fyzická, viditelná činnost, které je třeba se zhostit, aby se stávající stav věcí posunul blíže k dokončení. Pokud máme přesnou představu o dalším kroku, máme na výběr ze tří možností, jak se k němu zachovat:

- Vykonat
- Delegovat
- Odložit

Vykonáme každý krok, který zabere méně než dvě minuty. Krok s vyšší časovou náročností můžeme delegovat na někoho jiného. Jestliže není možnost krok delegovat, tak ho budeme muset uložit do seznamu dalších kroků. K lepšímu pochopení poslouží diagram na obrázku 2.1.



Obrázek 2.1 Diagram fáze zpracování

2.2.2.3 Uspořádání

Fáze uspořádání pracuje s kategoriemi, které vznikly během zpracování.

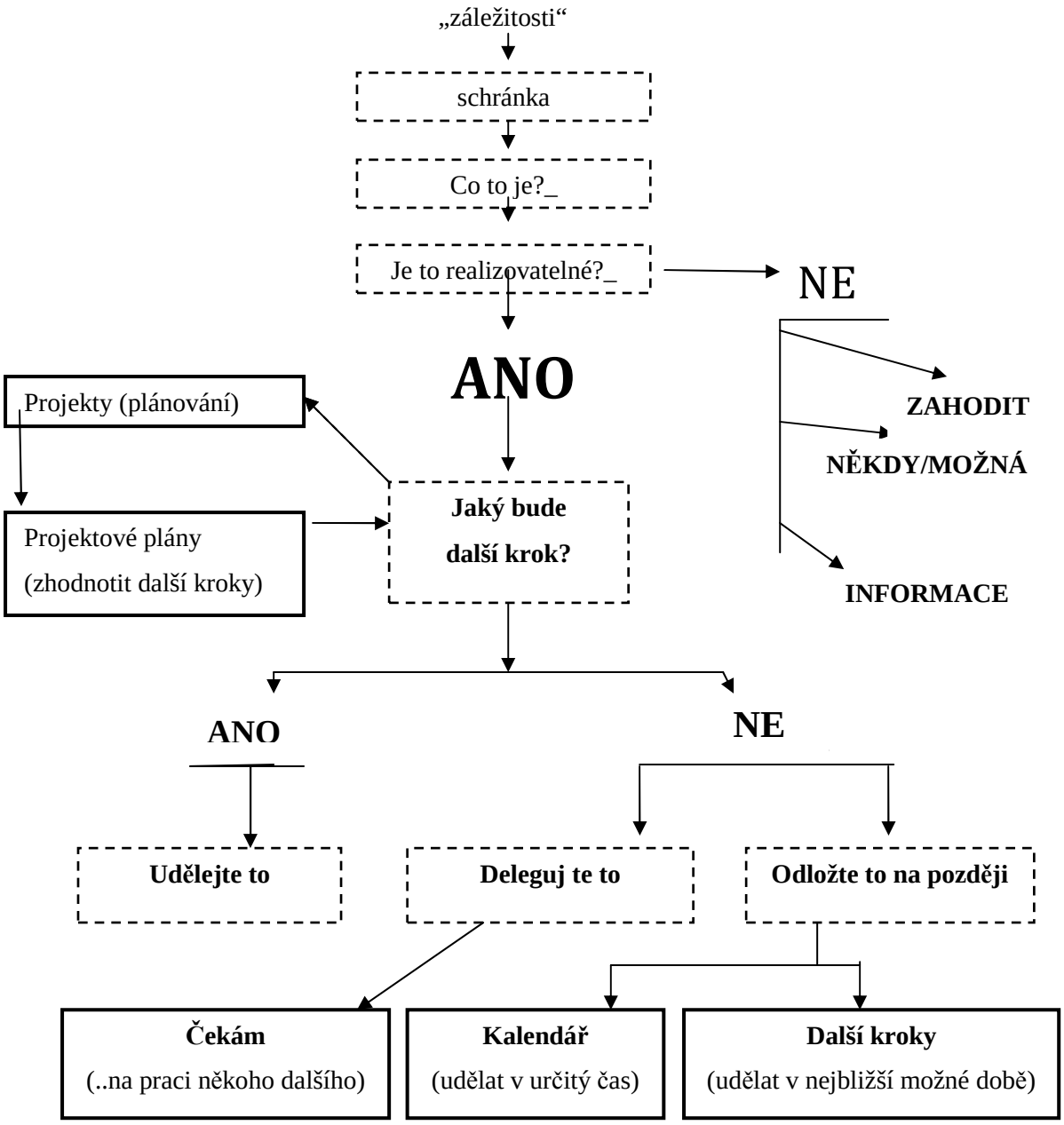
1. Seznam projektů
2. Projektové plánování
3. Zahodit
4. Někdy-Možná
5. Archív
6. Kalendář
7. Čekám na
8. Další kroky

Pro správné pochopení seznamu projektů si musíme uvést definici tohoto termínu podle GTD. Projekt je definován jako každý očekávaný výsledek, který vyžaduje více než jeden konkrétní krok. Je to skvělé upozornění na zbývající kroky, pokud se daná záležitost nedá splnit během jednoho kroku. Účel tohoto řešení je prostý, dostat z našeho mozku tuto záležitost, aby jí nemusel zabývat a narušovat naší koncentraci. Projekt zpracováváme postupně, dokud nedosáhneme kýženého výsledku a nejsme s ním maximálně spokojeni. V tomto okamžiku je náš projekt splněn.

Kalendář slouží k uchování kroků, jež mají pro své splnění přesně vymezené datum nebo čas. Dále si můžeme poznamenat užitečné informace na daný den. V žádném případě si však nesmíme poznamenat seznam úkolů na daný den. Podle většiny metod pro organizaci času bychom si měli pro daný den vytvořit seznam úkolů. Bohužel tento návyk se v praxi ukázal jako milný. Většinou se stalo, že uživatel nestihl v daný den splnit všechny úkoly a musel je přepsat na další den. GTD nám nabízí alternativu v podobě seznamu dalších kroků, kde toto neefektivní přehazování úkolů na další den odpadá.

Nerealizované položky jsou dvojího typu: k zahození a k dozrání. S prvním typem operace jednoduchou položku vyhodíme z našeho systému, že již dále nikde nebude figurovat. K dozrání položku zařadíme do kategorie „Někdy-Možná“. Jedná se o ty projekty, na něž se momentálně nemůžeme zaměřit, ale v budoucnu bychom se jim rádi věnovali.

Do archívu ukládáme položky s informativní hodnotou. Nevyžadují žádnou akci. Ukládáme je pro případ, abychom mohli potřebnou informaci snadno vyhledat. Fáze uspořádání můžeme vidět na obrázku 2.2.



Obrázek 2.2 Diagram fáze uspořádání

2.2.2.4 Hodnocení

Nejlépe jednou týdně bychom si měli vyhradit čas na tyto čtyři body.

- Sebrání a zpracování všech „věcí k vyřízení“
- Revizi systému
- Aktualizaci seznamů
- Ujasnění, aktualizaci a doplnění

Pomůže nám to k dodržování námi nově získaným návykům. Odstraníme nepořádek, který vznikl během uplynulé doby. Náš vybudovaný systém k organizaci času se stane kompaktnějším. Budeme mu více důvěřovat.

2.2.2.5 Provádění

Jaký krok bychom mohli provést. To záleží jen na nás. David Allen navrhuje rozhodnout se podle čtyř kritérií:

1. Kontext
2. Dostupný čas
3. Dostupná energie
4. Priorita

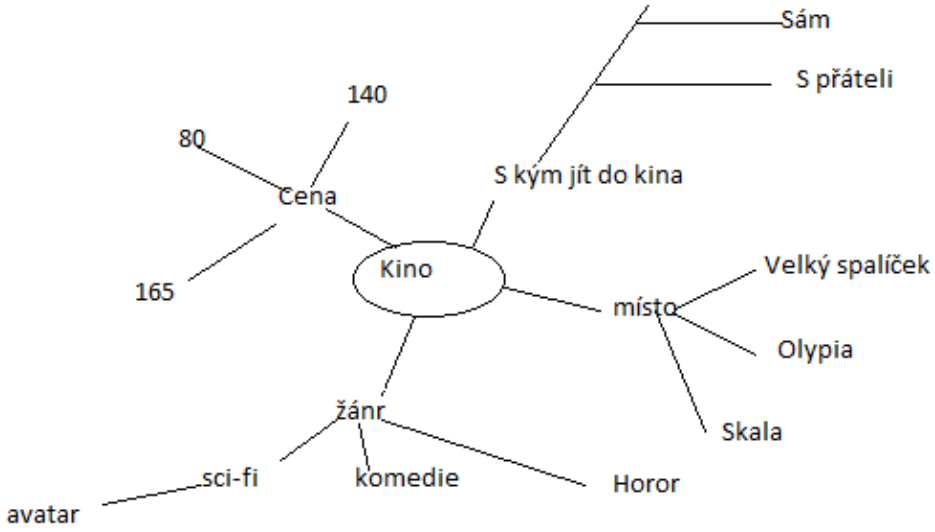
Kontext je ve většině případů fyzické místo, osoba či věc. Které je potřeba k vykonání daného kroku. Dostupný čas není třeba blíže popisovat neboť každému je jasné, že plníme pouze ty kroky, na něž máme dostatečné množství času. Krok se musí provést na jednou. Při plnění kroku se jedná o atomickou operaci. Zásadní kroky je dobré vykonat po ránu, když jsme ještě plni energie a odvedeme co nejlepší výkon. Prioritu určujeme podle toho, jak moc důležité je, aby se daný krok splnil.

2.2.3 Přirozené plánování

Nejlepší plánovač na světě je náš mozek. Bohužel starší metody pro organizaci času využívaly nepřirozené plánování. U přirozeného plánování využíváme vrozených pochodů mysli při plánování naších budoucích cílů. Náš cíl se organizuje pomocí pěti kroků:

1. Definice cíle a zásad
2. Představa o výsledcích
3. Brainstorming
4. Uspořádání
5. Určení dalších kroků

Následujících pět kroků si můžeme stručně popsat na praktickém příkladě návštěvě kina. Naše definice cíle a zásad je návštěva kina. Dále nám mozek vytvoří myšlenku, jak úžasný budeme mít pocit, až shlédneme film. To byla představa o výsledcích. Následuje obrovský proud myšlenek. Některé z nich nejsou ani realizovatelné. Náš mozek právě přepadl brainstorming. Neváhejte a všechny tyto nápady si poznamenejte. Můžete využít oblíbenou techniku mentálních map. S jejichž pomocí britský vědec Tony Buzan zachytil brainstorming do grafické podoby. Na obrázku 2.3 můžeme vidět mentální mapu popisovaného kina.



Obrázek 2.3 Mentální mapa

Existuje mnohem více způsobů záznamu brainstormingu. Jedno však mají společné, a to zaznamenat naše myšlenky mimo naši hlavu. Je jedno na jaké medium tento záznam provedeme. Po zapsání všech nápadů, začne náš mozek řadit jednotlivé položky. To jsme přešli ke kroku uspořádání. Organizujeme jednotlivé položky, které jsme získaly brainstormingem. To nám pomůže pochopit kostru našeho projektu. Poté již není problém doplnit chybějící části. K co nejlepší organizaci nám mohou pomoci tyto kroky:

- Rozpoznejte důležité nápady
- Roztřiďte je (podle jednoho nebo více kriterií)
  - Komponenty
  - Posloupnosti
  - Priority
- Rozpracujte do požadované úrovně podrobností

Finální fází plánování je další krok. Projekt se může skládat z více částí a v každé můžeme nalézt další krok. Nebo se nám také může stát, že na jednom kroku závisí osud celého projektu. Kolik dalších kroků potřebujeme ke splnění cíle je plně v naší kompetenci

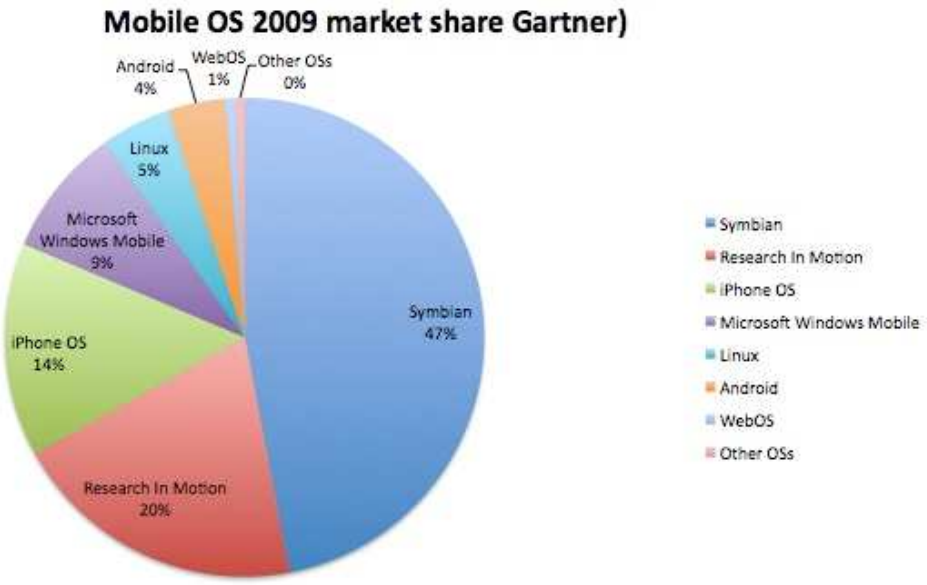
2.3 Shrnutí

Po přečtení této kapitoly by měl čtenář získat přehled o metodě GTD. Díky těmto znalostem by měl být schopen využít nástroj, který slouží pro podporu GTD. Jedná se o aplikaci pro Windows mobile. Ta bude důkladně popsána v následujících kapitolách.

3 Windows Mobile

3.1 Současná situace na trhu

Výrobci mobilních zařízení nabízejí své produkty s operačním systémem od firmy Microsoft. Mimo zmíněnou firmu působí na trhu nemalá konkurence. Jak lze vidět na koláčovém grafu, který představuje podíl na trhu s mobilními operačními systémy jednotlivých firem (Obrázek 3.1).



Obrázek 3.1 Podíl na trhu jednotlivých platforem [2]

Velice dobře si vede firma Nokia s operačním systémem Symbian. Firmy Google a Apple mají obrovský potenciál. To vše díky jejich neobvyklému pohledu při vývoji. Většina firem se zaměřuje hlavně na vývoj své platformy a vývoj cílových zařízení nechávají na jiných firmách. Díky tomu si potenciální zákazník může vybrat ze široké palety přístrojů. Naproti tomu firmy Apple a Nokia nabízejí své platformy pouze na přístrojích vyrobených v jejich továrnách. Má tu výhodu, že mohou dokonale přizpůsobit svoji platformu na určitý model zařízení.

Firma Microsoft se svým mobilním operačním systémem nezažívá nejlepší období. Přiznal to výkonný ředitel této společnosti Steve Ballmer, který byl znechucen z finančních výsledků Windows Mobile. Na základě těchto výsledků nechal přeorganizovat vývojový tým budoucí nové platformy Windows Phone 7. Lze tedy očekávat převratné změny v porovnání s předešlými verzemi. Někteří analytikové však předpokládají i možné ukončení vývoje mobilních platforem od firmy Microsoft. Tomu nemilému faktu nahrává obchodní politika výrobců zařízení, kteří již nemají tuto firmu jako hlavního dodavatele operačního systému do jejich zařízení. [3]

3.2 Konkurenční platformy

Během získávání dat o platformě Windows Mobile jsme se setkávali s informacemi o konkurenčních platformách. To nám pomohlo si vytvořit ucelenější obraz o problematice vývoje mobilních aplikací. Z toho důvodu si nejvýznamnější konkurenční platformy zmíníme.

3.2.1 Symbian OS

Setkáváme se s ním především v mobilních telefonech od firmy Nokia. Jeho předchůdce byl systém EPOC, kterým jsme mohli nalézt na zařízeních Psion. Symbian OS je otevřený operační systém, jehož zdrojové kódy jsou šířeny pod licencí Elipse Public Licence. Lze ho spustit jen na procesorech ARM. Uživatel může rozšiřovat funkčnost systému pomocí aplikací. Ty jsou však pěvně spjaty s verzí operačního systému. Nejnovější verze podporuje příjem digitálního vysílání. [4]

3.2.2 iPhone OS

Byl vyvinut firmou Apple. Dříve byl značen pod neoficiálním názvem OS X. Dnes je nainstalován zařízeních iPhone, iPad a iPod od již zmíněné firmy. Vychází z platformy MAC OS X. Je optimalizován pro přenosná zařízení iPhone OS je složen ze čtyř vrstev:

- core OS- jádro sytému
- vrstva Core Services- základní služby
- mediální vrstva
- cocoa Touch- uživatelské rozhraní

Uživatelé si iPhone OS oblíbili díky jeho ovládaní prsty a jednoduchému prostředí. Uživatel muže instalovat aplikace. Je však omezen ve výběru. Aplikaci si musí vždy kopit přes společnost Apple. Ona rozhoduje, jaké aplikace bude prodávat. Každá aplikace prochází schvalovacím procesem. [5]

3.2.3 Android

Tato platforma je nekratší dobu na trhu v porovnání zde již zmíněnými platformami. Android byl vyvinut firmou Google. Vychází z Linuxového jádra verze 2.6. Google prosazuje otevřené standarty v segmentu mobilních zařízení. Zdrojové kódy této platformy jsou šířeny pod licencí Apache free software and open source. Android má před sebou slibnou budoucnost. Neustále zvyšuje svůj podíl na trhu. [6]

3.3 Historie

Microsoft vytvořil první verzi Windows Mobile v roce 2000. Během následujících 10 let přišel s mnoha vylepšeními. Nejnovější zařízení obsahují verzi Windows Mobile 6.5. Tato verze je poslední s rodiny Windows Mobile. Systém Windows Phone 7 bude již zcela nová vývojová větev. Z toho důvodu aplikace vytvořené pro starší Windows Mobile nebudou kompatibilní s Windows Phone 7. Seznam verzí podle data vydání:

1. Pocket PC 2000
2. Pocket PC 2002
3. Windows Mobile 2003, Windows Mobile 2003 SE
4. Windows Mobile 5
5. Windows Mobile 6
6. Windows Mobile 6.1
7. Windows Mobile 6.5
8. Windows Phone 7

3.4 Vlastnosti

3.4.1 Cílová zařízení

V dnešní době je největším výrobcem mobilních zařízení s operačním systémem Windows mobile firma HTC. Každý rok přichází s nabídkou nových modelů. Je těžké definovat parametry standardního zařízení. Z toho důvodů si zde pro orientaci uvedeme hardwarovou vybavenost jednoho zařízení.

HTC Touch HD je osazen procesorem QualComm® MSM 7201 A 528 MHz (Obrázek 3.2). Operační paměť dosahuje velikosti 288 MB. Paměť k uchování dat má velikost 512MB. Můžeme ji rozšířit pomocí paměťových karet microSD. Integrovaný GPS modul je dnes samozřejmostí, slouží k určení polohy uživatele. Pro přenos dat má integrovaná tyto rozhraní Bluetooth, Wi-Fi (IEEE 802.11b/g) a mini USB 2.0. K zachycení obrazu nám slouží 5 megapixelový integrovaný digitální fotoaparát. Všechny informace se zobrazují na 3,8 palcový dotykový TFT-LCD displej s rozlišením WVGA. Energii k provozu těchto komponent získává z lithium-iontové polymerové baterie s kapacitou 1350 mAh. [8]

Zařízení, která jsou vybavena tímto hardwarem, mohou poskytovat pokročilé funkce. Proto je všeobecně označujeme termínem smartphone.

Obrázek 3.2 HTC Touch HD [8]

3.4.2 Prostředí Windows Mobile

V následujících větách si popíšeme obecné vlastnosti verzí od Windows Mobile 6.1 do Windows Mobile 6.5, jejichž funkčnost se moc neliší a stále jsou hodně používány mezi uživateli. Nejvýraznější rozdíl najdeme v ovládání. Windows Mobile 6.5 má uzpůsobeno uživatelské rozhraní pro dotykové ovládání prsty. Obě verze podporují spouštění více aplikací najednou. Uživatel si může přepínat mezi běžícími aplikacemi.

Windows Mobile obsahuje řadu vynikajících nástrojů usnadňujících práci na zařízení. Windows Mobile Device Center je program určený k synchronizaci s osobním počítačem. Nahradil dříve rozšířený ActiveSync. Uživatel nemusí řešit, jestli pracuje na počítači nebo s mobilním zařízením. Dokumenty vytvořené pomocí Microsoft office se automaticky synchronizuji. Uživatel má vždy svá aktuální data stále po ruce. Ke standardní výbavě operačního systému patří multimediální přehrávač Windows Media Player.

Mobile office slouží ke kancelářské činnosti, obsahuje Word Mobile, Excel Mobile, Powerpoint Mobile a Onenote Mobile. Zmíněné programy jsou funkčně omezeny porovnáním s verzí pro osobní počítače. Samozřejmostí je podpora bezdrátových komunikačních technologií Wi-Fi a Bluetooth. Chybí-li nám jistá funkčnost ve Windows Mobile, není problém si doinstalovat aplikaci, která nám ji doplní.

3.4.2.1 MyPhone

Firma Microsoft pociťuje obrovský tlak konkurence, a proto přichází s MyPhone. Jedná se o synchronizační službu. Abychom mohli, synchronizační službu využívat musíme si do svého mobilního zařízení stáhnout aplikaci Microsoft MyPhone. Odkaz k získání aplikace nám bude zaslán prostřednictvím SMS. Ta se nám zašle po zadání našeho telefonního čísla na domovské webové stránce aplikace http://myphone.microsoft.com. Po úspěšné instalaci aplikace. Minimálně při jejím prvním spuštění po nás aplikace bude chtít náš Windows Live ID účet k přihlášení. Pokud ho

nemáme, aplikace nás odkáže na webovou stránku, kde si ho budeme moci vytvořit. Po úspěšném přihlášení nás uvítá hlavní nabídka s možnostmi výběru synchronizace položek. Nabídka je opravdu rozsáhlá.

Data určená k synchronizaci nám aplikace přenese pomocí sítě internet na server. Zde si je můžeme prohlížet pomocí webového prohlížeče. Data lze různě editovat popřípadě mazat. Tyto změny se po synchronizaci objeví v mobilním zařízeni. Velice užitečná věc je synchronizující kalendář se stejnými funkcemi, které známe z Windows Mobile. Webové rozhraní také nabízí určení polohy mobilního přístroje. Tato funkce může byt užitečná v případě jeho ztráty. Jediná nevýhoda této služby je omezený datový prostor pro synchronizaci. Nyní jeho velikost činí 200MB. Jedná se o skvělý nástroj pro uložení důležitých dat, usnadňující migraci na nové zařízení.

3.4.2.2 Windows Marketplace

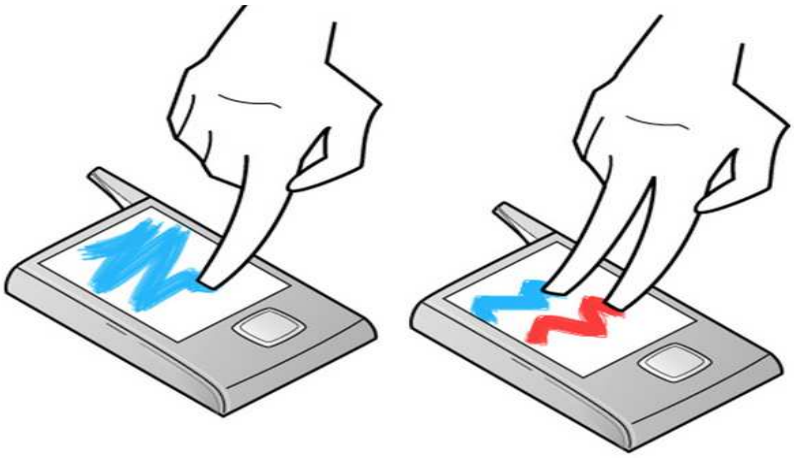
Windows Marketplace je centrální internetový obchod s aplikacemi. Před tímto počinem byly aplikace pro Windows Mobile roztříštění po různých internetových portálech. Uživatel nejdříve musel složitě požadovanou aplikaci vyhledat. Nyní si stačí nainstalovat stejnojmennou aplikaci. Prostřednictví Windows Marketplace si uživatel může stahovat aplikace. Podporuje vyhledávání podle různých kritérií například oblíbenosti, ceny a kategorie.

Windows Marketplace představuje skvělý systém, jak nabídnout aplikaci zákazníkovi. Plyne z toho také výhoda pro vývojáře. Ti se nemusejí starat o distribuci svých aplikací. Na druhou stranu aplikace, které chceme distribuovat tímto způsobem, musí projít schvalovacím procesem firmy Microsoft. Ta rozhodne, jestli aplikaci zařádí do katalogu Windows Marketplace. Vývojář dostává 70% z ceny prodeje aplikace. Tento obchodní model byl převzat od firmy Apple.

3.4.3 Ovládání Windows Mobile

Většina nejranějších zařízení se ovládala pouze hardwarovými tlačítky. Bylo to velice pomalé. Uživatel se musel postupně se promačkávat přes všechny systémové nabídky. V průběhu doby se začali prosazovat dotykové obrazovky, k nimž se dodával stylus. Je to taková tužka zakončená kulatým hrotem. Usnadňuje nám ovládání aplikace. Jakmile se stylusem dotkneme vybraného místa na obrazovce, program na tento dotyk zareaguje, vyvoláním příslušné události. Aby systém věděl přesnou polohu našeho dotyku, při jeho prvním spuštění musíme provést kalibraci displeje. Ta se provádí pomocí programu, který je součástí systému.

Se zvětšováním plochy mobilní obrazovky mohou ovládací prvky zabírat více místa, a tak je možnost používat k ovládání prsty vlastní ruky. Někteří výrobci přestaly dodávat stylus ke svým zařízením, jelikož ovládání prsty přináší uživateli větší komfort řízení aplikace. Podle průzkumů si zákazníci přejí dotykové ovládání. Proto Windows Phone 7 bude podporovat multi-touch.(Obrázek 3.3). Tato technologie je schopna reagovat na více dotyků zaráz. Program je schopen poznat jestli se uživatel dotkl obrazovky jedním nebo více prsty.



Obrázek 3.3 Ovládání Multi-Touch

3.4.4 Změna orientace obrazovky a rozlišení

Windows Mobile podporuje změnu orientace obrazovky. Ta se provádí na požadavek od uživatele nebo automaticky. Zařízení dokáže ve vybraných případech rozpoznat jaká orientace obrazovky je výhodná pro uživatele. Například v reakci na událost vygenerovanou G-senzorem nebo výsuvnou klávesnicí. G-senzor nám umožňuje rozpoznat, v jaké poloze se zařízení nachází a podle toho zvolí režim zobrazení. Při změně taky dochází ke změně souřadnicového systému. Začátek souřadnicového systému je vždy v levém horním rohu. U obrazovek s rozdílným poměrem stran máme na výběr z režimů *Landscape* a *Potrait*. V jakém režimu se obrazovka právě nachází, zjistíme z poměru rozlišení horizontálního a vertikálního. Hodnoty poměru jednotlivých režimů:

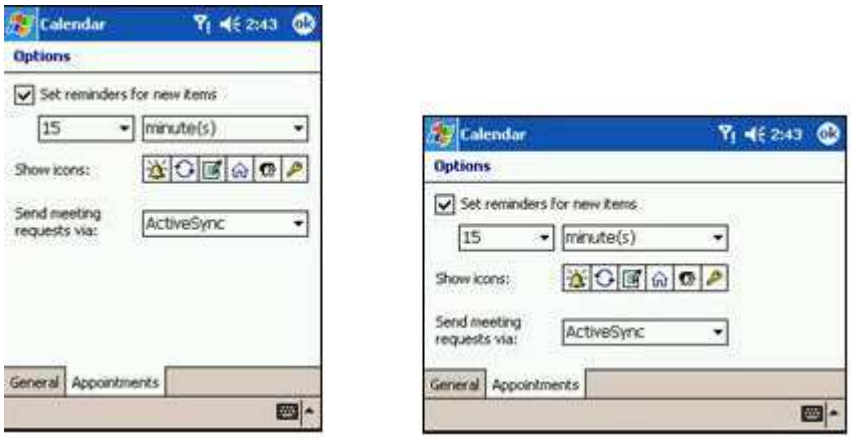
- *Potrait* < 0.8
- 0.8 ≤ *Simple* ≥ 1.3
- *Landscape* > 1.3

V případě *Simple* režimu se při změně orientace poměr stran nezmění nebo jen nepatrně. *Landscape* je zobrazení na délku obrazovky (Obrázek 3.4). *Potrait* je na výšku (Obrázek 3.4). Během těchto změn by aplikace měla umět přizpůsobit ovládací prvky danému režimu. Mohl by nastat případ, že ovládací prvky se vykreslí mimo obrazovku. To by způsobilo nefunkčnost programu. Uživatel by nemohl program řádně ovládat (Obrázek 3.6).

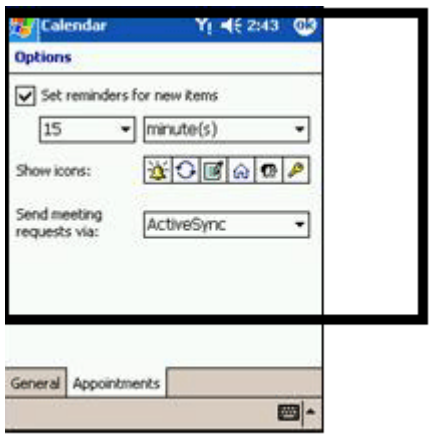
Rozlišení je vlastnost zařízení, která udává počet zobrazených bodů na obrazovce. Tyto body se nazývají pixely. Většina zařízení se spadá do určitého zobrazovacího standardu. U Windows Mobile se nejčastěji setkáváme s těmito rozlišeními:

- QVGA (320×240)
- VGA (640×480)
- WGA (800×480)

Rozdíly v rozlišení stěžují vývoj aplikací. Programátor musí vytvořit program, který se dokáže přizpůsobit danému rozlišení. Některé návrhy programů je nemožné přizpůsobit na nižší rozlišení.



Obrázek 3.4 na levé straně Potrait na pravé Landscape [10]



Obrázek 3.5 Nepřizpůsobení ovládacích prvků Landscape modu [10]

3.5 Vývoj aplikací pro Windows Mobile

3.5.1 Windows Mobile Standard developer kit

Pro vývoj aplikací je připraven modul Windows mobile standart developer kit. Před instalací si musíme vybrat, pro jakou verzi Windows Mobile chceme vyvíjet aplikaci. Na základě tohoto rozhodnutí si stáhneme odpovídající verzi SDK. Například pro Windows mobil 6.1 je to Windows Mobile 6 SDK. Dále máme na výběr dvě edice *Professional* a *Standard*. Hlavní rozdíl v těchto edicích je podpora dotykového displeje v *Professional*. Jak můžeme vidět v následujícím seznamu. Ten byl převzat ze zdroje [11].

- **Windows Mobile 6 Standard SDK**
  - Windows Mobile 6 Standard (176×220 pixels 96 dpi)
  - Windows Mobile 6Standard Landscape QVGA (240×320 pixels 131 dpi)
  - Windows Mobile 6 Standard QVGA (320×240 pixels 131 dpi)
- **Windows Mobile 6 Professional SDK**
  - Windows Mobile 6 Classic (240×320 pixels 96 dpi)
  - Windows Mobile 6 Professional (240×320 pixels 96 dpi)
  - Windows Mobile 6 Professional Square (240×240 pixels 96 dpi)
  - Windows Mobile 6 Professional Square QVGA (320×320 pixels 128 dpi)
  - Windows Mobile 6 Professional Square VGA (480×480 pixels 192 dpi)
  - Windows Mobile 6 Professional VGA (480×640 pixels 192 dpi)

Vývojový modul obsahuje emulátory mobilních zařízení. Při testování aplikací je to nenahraditelná pomůcka. Bohužel v emulátoru se aplikace nechová vždy stejně jako na fyzickém zařízení. Modul podporuje vytváření instalačních balíčků. Modul se doinstaluje do vývojového prostřední Visual Studio. Jeho součástí je .NET Compact Framework.

3.5.2 Microsoft Visual Studio

Jedná se o vývojové prostředí od firmy Microsoft. První verze byla vydána v roce 1997 a od té doby prošla bouřlivým vývojem. V následujících odstavcích proto uvedu popis jen těch informací, které subjektivně považuji za důležité.

Pro psaní programu má vestavěný editor. Umožňuje programátorovi komfortní programování v podobě podpory zvýrazňování syntaxí a automatické doplňování zdrojového kódu. K odstraňování chyb ve vyvíjených aplikací slouží debugger, který pracuje jak se spravovaným kódem, tak se strojovým kódem a může být použit pro ladění aplikací psaných v jakémkoliv jazyce.

Grafické designéry nám pomáhají s vývojem aplikací. Ve Visual Studiu se můžeme setkat s WinForms Designerem nebo Class Designerem. Při návrhu aplikace pomocí WinForms Designeru si uživatel může vybrat z palety nástrojů před připravené ovládací prvky, které chce implementovat. Class Designer obsahuje grafické rozhraní pro vytváření diagramu tříd v UML. Z následných UML diagramů můžeme vygenerovat třídy v libovolném programovacím jazyce. Nebo naopak ze zdrojového kódu získáme diagramy UML.

Vyvíjenou aplikaci můžeme rozdělit do více projektů, které lze navzájem propojovat pomocí jmenných prostorů. Projekty se seskupují pod strukturami jménem Solution.

Rozšiřitelnost Microsoft Visual Studia je dána především možností instalace různých zásuvných modulů, které jsou k získání na adrese http://visualstudiogallery.msdn.microsoft.com

Dokumentace k zdrojovému kódu se vytváří pomocí základních dokumentačních xml značek. Každý řádek dokumentace začíná třemi lomítky (///).

Při stisknutí třikrát za sebou znaku lomítka nám Visual Studio automaticky vygeneruje základní xml značky, které můžeme doplnit o náš text dokumentace. Jestliže nám nevyhovuje základní generování dokumentace, lze si doinstalovat modul jménem GhostDoc [12]. Jedná se o velice propracovaný nástroj pro vytváření dokumentace ve formátu xml. Existuje obrovské množství nástrojů, které tento formát umí převést na jiný námi požadovaný. Například program SandCastle nám xml formát převede do html či LaTexu.

3.5.3 Microsoft .NET Framework

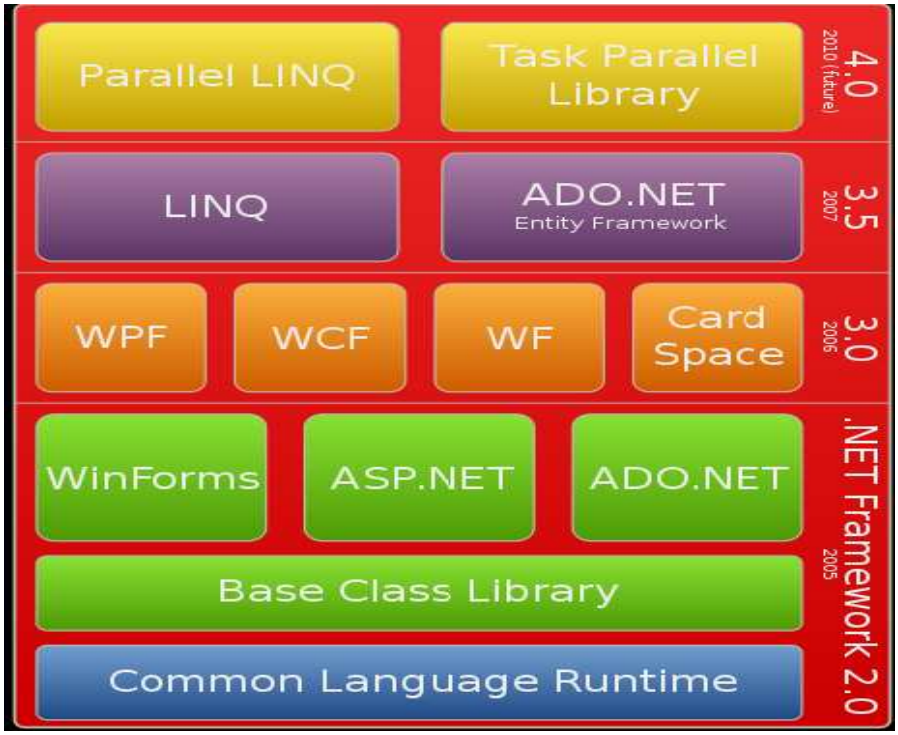
Tento název zastřešuje soubor technologií v softwarových produktech, které tvoří celou platformu. Základní komponentou je Microsoft .NET Framework. Jedná se o prostředí, v němž je spuštěna aplikace, které zároveň obsahuje potřebné knihovny pro správný běh. .NET platforma se díky vývoji rozrostla do takové míry, že má v sobě zabudované technologie téměř pro jakýkoliv segment vývoje v IT. Platforma je dostupná pro rozdílné typy operačních systémů

Edice pro danou platformu operačního systému:

- *Microsoft .NET Framework*- pro personální počítače s operačním systémem od firmy Microsoft.
- *Microsoft .NET Compact Framework*- pro mobilní zařízení s operačním systémem od firmy Microsoft.
- *Microsoft .NET Micro Framework*- pro zařízení kategorie embled, která disponují velmi omezenými výpočetními zdroji.
- *Mono*- podporován firmou Novell, jedná se o open source. Řešení pro operační systémy Unixového typu.

3.5.3.1 Dělení Frameworku .NET podle verze

Další dělení platformy je podle stáří verze. Nejstarší měla velmi omezenou množinu funkčnosti, například chyběly v ní dnes již zcela běžné generické seznamy. Verze jsou číslovány vzestupně. První nesla označení .NET Framework 1.0 a byla vydána v roce 2000. Dříve každá nová verze vyšla zároveň s novým vývojovým prostředím Microsoft Visual Studio. Nyní již toto pravidlo neplatí. U Microsoft Visual Studia 2008 si můžeme zvolit verzi Frameworku .NET pod kterou chceme projekt vyvíjet. Na obrázku můžeme vidět postupné přidávání nových knihoven.



Obrázek 3.6 Architektura .NET a změny v nových verzích. [13]

3.5.3.2 .NET Compact framework

Pro Windows Mobile máme připravenou speciálně upravenou edici této platformy. Její název je .NET Compact Framework. Jedná se o podmnožinu .NET Frameworku, která je optimalizována pro mobilní zařízení a jejich nižší výpočetní výkon.Aktuální verze nese číselné označení 3.5.

3.5.3.3 Architektura platformy Microsoft .NET

Základní architektonickou strukturu platformy .NET (Obrázek 3.7) tvoří čtyři komponenty:

1. *Bázová knihovna tříd* (FCL, Framework Class Library)
2. *Virtuální exekuční systém* (CLR, Common Language Runtime)
3. *Společný typový systém* (CTS, Common Type Specification)
4. *Společná jazyková specifikace* (CLS, Common Language Specification)

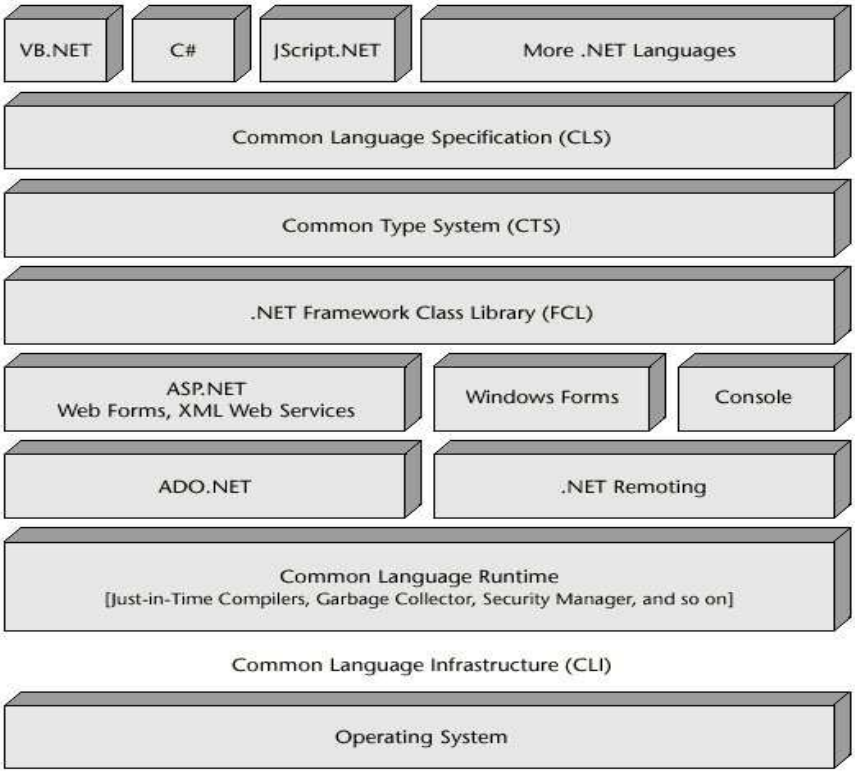
Bázová knihovna tříd obsahuje již předem připravené třídy, delegáty, rozhraní a další softwarové entity. Jde o tzv. prefabrikované bloky kódy. Vývoj aplikace tím čím dál více připomíná skládání skládačky. Kde jednotlivé části představují prefabrikované bloky kódu. Jsou to například operace s bitmapovými mapami, generickými seznamy, databázemi, síťovým rozhraním atd. Z tohoto důvodu se vždy nejdříve vyplatí podívat, jestli požadovanou funkčnost již nenabízí některý z prvků v knihovně. Nemusíme vyvíjet to, co již je dávno hotovo a připraveno k použití.

Virtuální exekuční systém je zodpovědný za správný běh aplikací .NET. Mezi nimi je silná vazba. CLR řídí vykonávaní .NET aplikací. Ta neobsahuje instrukce strojového kódu. Místo toho leží uvnitř aplikace .NET fragmenty jazyka známého jako Microsoft Intermediate (zkráceně MSIL, případně CIL- Common Intermediate Language). MSIL jazyk je objektový jazyk nízké úrovně, mohli

bychom jím psát aplikace pro .NET, ale za cenu velkého snížení komfortu v porovnání s ostatními jazyky. Abychom mohli aplikaci vykonat, potřebujeme převést MSIL jazyk na strojové instrukce, které umí vykonat procesor. Tento nástroj je nazýván Just In Time (JIT) kompilátor.

Společný typový systém slouží k bezproblémové interoperabilitě mezi různými jazyky. Definuje všechny společné požadavky, jež musí splňovat datové typy. Zároveň se musely standardizovat jazyky a jejich kompilátory. Byla vytvořena základní kolekce požadavků, která dostala název Společná jazyková specifikace (CLS). To nám zaručuje, že aplikace napsané v různých jazycích platformy Microsoft .NET budou moci vést mezi sebou informační dialog. V minulosti nastával problém při změně programovacího jazyka, musely se přepisovat moduly již dříve napsané v jiném programovacím jazyce. CLS splňují například tyto jazyky C#, Visual Basic 2008 a C++/CLI.

Podkapitola čerpala informace ze zdrojů [13] [14] [15].



Obrázek 3.7 Architektura Microsoft .NET podrobnější pohled [13]

3.5.4 Jazyk C#

Jedná se o hybridní objektově orientovaný jazyk. Podporuje všechny stěžejní principy objektově orientovaných jazyků, k nimž patří abstrakce, zapouzdření, ukrývání dat, ukrývání implementace, dědičnost a polymorfismus. Byl standardizován organizacemi ISO (International Organization for Standardization) a ECMA (European Computer Manufactures Association). Jedná se o dokumenty

ECMA:334 C# a Language Specification a ISO/IEC 23270 Information Technology Programming Languages C#.Vyznačuje se vysokou pracovní produktivitou.
Vděčí tomu především za optimálně na projektovanou jazykovou specifikaci. Zápis algoritmů do jazyka C# je vykonáván rychleji a s menší námahou ve srovnání s jinými programovacími jazyky. Vychází z jazyka C++.Na rozdíl od něho jazyk C# má méně košatou jazykovou specifikaci. Eliminuje to spletité konstrukce, v kterých mnohdy programátor udělal zbytečné chyby. Podporuje objektově orientované paralelní programování, které vede ke zvýšení efektivnosti u dnešních více jádrových procesorů. [14]

S jazykem C# jsme se poprvé mohli setkat v produktu Visual C# .NET, jenž byl součástí Microsoft Visual Studia .NET. Jednalo se o verzi C# 1.0. Podporovala pouze základní vlastnosti objektového programování. Aktuální verze je nyní C# 3.0. Ta podporuje tyto vlastnosti:

- Částečné a statické třídy.
- Iterátory.
- Anonymní metody pro pohodlnější užívání delegátů.
- Nullovatelné hodnotové typy a operátor koalescence.
- Language Intergrated Query
- Prvky funkcionálního programování
- Inicializátory objektů a kolekcí
- Klíčové slovo var (netypovost)
- Výrazové stromy
- Anonymní třídy

Vývoj tohoto jazyka neustále pokračuje a v Microsoft Visual 2010 Studiu je již obsazená verze C# 4.0.

3.5.5 XML Jazyk

Extensible Markup Language je značkovací jazyk. Byl vyvinut a standardizován konsorciem W3C. Uživatel si sám definuje značky, a tím zvyšuje obsah informací. Značky je navíc možné definovat v DTD souboru. Pomocí něho lze jednoduše zjistit, jestli vytvořený xml soubor odpovídá námi vytvořeným pravidlům. XML jazyk je vhodný přenosový formát mezi rozdílnými platformami. [18]

3.5.6 Serializace a Deserializace XML

Serializace je proces, při kterém je objekt nebo graf objektů převeden na lineární posloupnost bajtů. Framework .NET 3.5 podporuje tři druhy výstupu, a to binární, SOAP a XML. Nás bude zajímat pravě XML výstup z již zmíněných důvodu v příchozím textu.

Deserilizace nám vytvoří živý objekt serializovaného objektu. V programovacím jazyce se nachází funkce, která nám z xml souboru vytvoří objekt, který přetypujeme podle třídy. [18]

Serilizovaný objekt do výstupného xml souboru muže vypadat jako na následujícím příkladu z aplikace GTD:

```
<?xml version="1.0" encoding="utf-8" ?>
<ArrayOfProjekt xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<Projekt>
 <ListCS>
  <CategoryStep>
  <Name>Aktuální</Name>
    <ListStep>
     <Step>
      <Name>uklid</Name>
      <Context>Doma</Context>
      <Note />
      <Done>false</Done>
     </Step>
  </CategoryStep>
 </ListStep>
</ListCS>
<Active>true</Active>
<Name>byt</Name>
<Note />
</Projekt>
</ArrayOfProjekt
```

3.5.7 Instalační balíček .cab

Pomocí Microsoft Visual Studio 2008 lze vytvořit kabinetní soubor. Jedná se o instalační soubor pro operační systém Windows Mobile. Do něj lze přidat jakýkoliv soubor, popřípadě nakonfigurovat ho tak, aby se nám při instalaci data nakopírovala do námi požadovaného místa v systému. Například potřebujeme-li vytvořit ikonu zástupce v programech. Kabinetní soubor je vždy zakončen příponou .cab.

4 Návrh aplikace GTD

Vnikla potřeba vytvořit prostředek k co nejefektivnější správě GTD agendy. Uživatelé metody GTD nejčastěji používají běžné prostředky pro její agendu. Psací potřeby bohatě vystačí k této činnosti. Nenabízejí však takový komfort záznamu, jako když použijeme elektronické medium. Jestliže naši myšlenku posuneme ještě dále, zjistíme, že nejvýhodnější by bylo tento záznam uchovat v zařízení, které máme stále po ruce. Tyto výhody právě poskytuje mobilní zařízení.

Hlavní koncepce aplikace GTD se zaměřuje na uspoření času při agendě metodologie GTD. Tím se tento nástroj stává bezkonkurenční oproti konvenčním způsobům.

Aplikace musí být schopna uchovávat základní stavební prvek GTD, kterým je krok. U něhož si musíme uchovat jisté informace. Například jsou to název, kontext, delegování, poznámka a datum odložení. Tyto kroky musí byt přiřazeny ke společnému cíli, což je projekt. K projektu potřebujeme přiřadit poznámku a přesouvat ho do určitých kategorií podle toho v jakém stavu se nachází. Výběr dalšího kroku musí trvat co nejkratší dobu. Při návrhu došlo na otázku, jak se vypořádat s perzistencí dat aplikace po jejím ukončení. Pro zachování co největší kompatibility byla zvolena perzistence pomoci serilizace a uložení do xml souboru.

Již při návrhu jsme si uvědomili výhody rozdělení aplikace do nezávislých vrstev. Po dekompozici návrhu nám vznikly tři vrstvy:

- **Datová**
- **Aplikační**
- **Prezentační**

Usnadnilo nám to možnosti rozšíření v průběhu procesu návrhu a implantace. Některé části jsme mohli vyvíjet nezávisle na sobě, což bylo výhodné z hlediska úspory času. Jestliže jsme narazili v jedné části na problém, který nešel vyřešit ihned. Mohli jsme řešit jinou část a vývoj neustrnul.

Museli jsme vzít v potaz i jisté omezení v podobě cílového zařízení. Není možné očekávat stejný uživatelský komfort, na jaký jsme zvyklí z osobních počítačů. To však neznamená, že se nebudeme snažit o co největší přiblížení k tomuto standartu.

5 Implementace aplikace GTD

Pro implementaci aplikace GTD jsem se rozhodl použít programovací jazyk C#. Jako vývojové prostředí jsem zvolil Microsoft Visual studio 2008. Pro vyšší efektivitu vývoje jsem použil platformu Microsoft .NET, konkrétně Compact Framework .NET 3.5.

5.1 Třívrstvá Architektura

Již z návrhu víme, že jsme aplikaci rozdělili do tří vrstev Datové, Aplikační a Prezenční. Vrstvy jsou na sobě nezávislé a mezi sebou komunikují přes rozhraní. Aplikační vrstva tvoří spojovací článek mezi vrstvou Datovou a Prezenční. Výhoda je především ve znovu použitelnosti již vytvořeného zdrojového kódu.

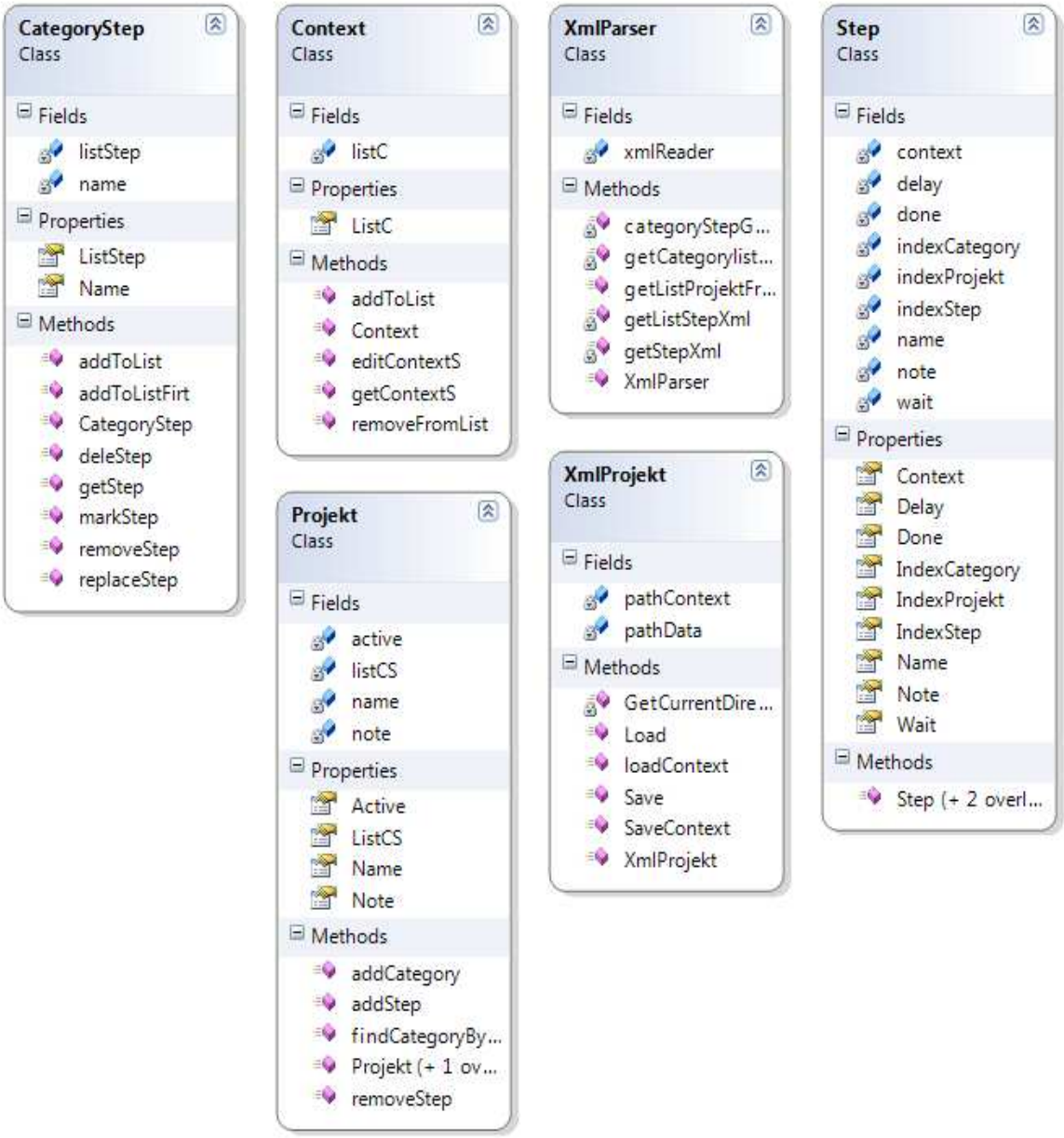
5.1.1 Datová vrstva

V aplikaci je pojmenována *DataLayer*. Z pohledu třívrstvé architektury lze na ni nahlížet jako na nejnižší vrstvu. Pracuje s daty, se kterými aplikace operuje. Obsahuje třídy zobrazené na obrázku 5.1. Třída *Step* slouží k uchování informací o daném kroku. Třída *CategoryStep* v sobě uchovává seznam kroků přiřazených do této kategorie a její název. Každá kategorie spadá pod určitý projekt a k tomuto účelu máme implantován seznam kategorii ve třídě *Projekt*. Třída *Context* obsahuje seznam názvů kontextů.

5.1.1.1 Perzistence dat pomocí XML

K zajištění perzistence dat byla využita serializace. Program musí vytvořit soubor xml v adresáři *Data*. Ten se nachází ve stejném adresáři jako soubor pro spuštění aplikace. Uživatel si během instalace sám rozhoduje, kam aplikaci nainstaluje. Abychom věděli, kde ji nainstaloval. Musíme zjistit cestu k souboru v operačním systému. K tomu nám poslouží funkce *GetCurrentDirectory()*. Ta má za návratovou hodnotu cestu od kořenového adresáře k aktuálnímu adresáři, z něhož je spuštěn projekt. Jedná se o tzv. absolutní cestu. Abychom zjistili úplnou cestu do adresáře data, je třeba zkombinovat absolutní a relativní cestu, k tomu máme funkci *path.combine()*. Nyní již známe cestu potřebnou k zápisu xml streamu do souboru.

Bohužel verze Compact .Net Frameworku nepodporuje u deserilizace generické datové typy. Proto jsme byli nuceni si dopsat vlastní XML parser, jehož implementaci popisuje třída *XMLParser*. U ní jsme použili třídu *XmlTextReader*. Třída podporuje procházení XML souboru po uzlech zavoláním metody *read()*. Má také definovány metody pro zjištění informací o vlastnostech uzlu. Ty jsou název uzlu, hodnota uzlu, typ uzlu atd.



Obrázek 5.1 Diagram tříd v Prezenční vrstvě

5.1.2 Aplikační vrstva

V aplikaci ji najdeme pod názvem *BusinessLayer*. Slouží k manipulaci s daty. K tomuto účelu jsou do ní implantovány dvě třídy *ContextOperation* a *ListProjekt*. Implantovány jsou funkce, které slouží prezentační vrstvě. Naopak tato vrstva využívá funkce datové vrstvy. Obě třídy mají implantovanou rozsáhlou množinu funkcí. Třída *ContextOperation* pracuje se seznam kontextů.

5.1.2.1 Třída ListProjekt

Má implantovány metody pro operace se seznamem projektů. Z Prezenční vrstvy dostává požadavky k naplnění daty kontejneru *ListViewItem*. K tomu slouží metody s prefixem *CreateListView*. Pomocí nich můžeme vytvořit seznam projektů, kategorii a kroků. Uchovává nám informaci, v jakých datech se uživatel pohybuje. V našem případě jsou to seznamy navzájem zanořené.

Máme tři úrovně možného zanoření seznamů. Na vrcholu je seznam projektů, do něhož je zanořen seznam kategorií, ve kterém zanořujeme seznam kroků. Proto vznikla potřeba pro vytvoření mechanismu, podle kterého by program vždy věděl, v jaké části dat se nachází. Řešením je poznamenávat si cestu uživatele při výběru dat pomocí indexu. K adresaci kroku nám stačí tři indexy.

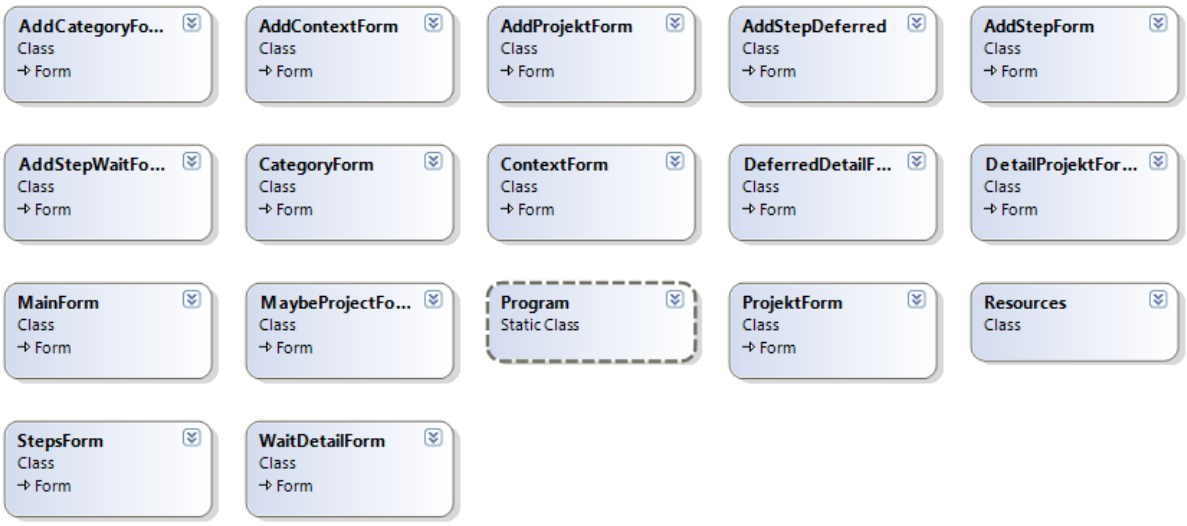
Ve třídě *ListProjekt* jsou implantovány pomocí třech proměnných s prefixem *Index*. *IndexProjekt* nám určuje pozici v seznamu projektů. *IndexCateogry* nám určuje pozici v seznamu kategorií. Poslední *IndexStep* nám určuje pozici v seznamu kroků. Tento mechanismus využívá mnoho metod s operacemi se seznamy.

Obstarává vytváření seskupování kroků dle určitých parametrů. Zde nastal problém s přidáváním objektu do kontejneru. Jazyk C# na rozdíl od jazyka Javy se chová jinak. Objekt do kontejneru zkopíruje celý, místo aby jen přidal odkaz na tento objekt. Tím vznikne problém při vytváření nového seznamu kroků z již vytvořených seznamů. Požadujme, aby se změny provedené v novém seznamu kroků provedli v těch seznamech, z nichž byl nový seznam vytvořen. Proto jsme byli nuceni naprogramovat metodu *refleshListProjekt*. Ji zavoláme vždy po ukončení operace se složeným seznamem kroků. Stálé seznamy se nám aktualizují. Zachováme tím datovou konzistenci.

5.1.3 Prezentační vrstva

Najdeme ji v aplikaci pod názvem *PressentationLayer*. Jejím hlavním účelem je interakce s uživatelem. Obsahuje velké množství tříd se společnou vlastností, nabídnout uživateli co nejkonformnější rozhraní pro komunikaci s ním (Obrázek 5.2). Jak již bylo zmíněno, tato vrstva využívá služeb aplikační vrstvy pro vykonání požadavků od uživatele nebo jen k naplnění datových kontejnerů sloužících k zobrazení dat uživateli. Během implantace jsme využili předem připravený komponent.

Následující podkapitoly čerpají informace ze zdrojů [19] [20].



Obrázek 5.2 Diagram tříd Aplikační vrstvě

5.1.4 Windows Forms

Je jedna z částí Microsoft .NET. Obsahuje připravené bloky uživatelského rozhraní, které se skvěle hodí při vývoji aplikace.

5.1.4.1 Třídy z knihovny Windows Forms

Téměř všechny třídy v prezentační vrstvě jsou potomky třídy *Form*. Tato třída určuje vlastnosti klientské části, což je povrch aplikace k zobrazování grafických informací. Mimo to definuje události, k nimž dojde, například při kliknutí na klientskou část. Na tento povrch můžeme nanášet ovládací prvky. V našem případě se jedná například o tlačítka, textová pole, výběrová pole atd. Jedno mají společné, jsou potomkem třídy *Control*.

Další důležitou vlastností potomků třídy *Control* je způsob umisťování ovládacích prvků. Můžeme je umísťovat absolutně nebo relativně. Kdyby všechna cílová zařízení měla stejné vlastnosti, například rozlišení obrazovky a její velikost, tak by nám nic nebránilo použít absolutní určování pozice. V reálném prostředí se setkáváme s rozdílnými vlastnostmi. Abychom zajistili co největší komptabilitu, používáme relativní určování pozice. Vývojové prostřední .NET Compact Framework 3.5 nám nabízí dvě možnosti *Anchor* a *Dock*.

Anchor znamená v překladu kotva. Ukotvený prvek si vždy bude udržovat stejnou vzdálenost od okraje, k němuž jsme ho ukotvili. Mechanismus ukotvení nám umožňuje vyšší kompatibilitu se zařízeními, jejichž obrazovka má rozdílné rozlišení a velikost.

Vlastnost *Dock* nám umístí vybraný prvek vždy k některému ze čtyř okrajů nebo můžeme vyplnit celý prostor.

Ovládací prvky

Slouží v aplikaci k interakci s uživatelem. Zajišťují vyvolání události na různé požadavky. Programátor může určit reakce na ně. Například přiřazením funkce k některé z událostí, která bude vykonána. V následujících odstávcích budou popsány prvky, jenž jsme v procesu implantace hojně využívali.

Label je neupravitelný popisek. Má vlastnost *TEXT*. Do ní ukládáme text, který chceme zobrazit. *Button* je komponenta která představuje tlačítko. Uživateli znázorňuje akci při kliknutí na něj. Dále je jeho hlavním účelem spouštění akcí. *CheckBox* komponenta dává na výběr zaškrtnutím, jestli položka platí nebo ne.

TextBoxt slouží k zadávání textu nebo uživatel může editovat text zobrazený pomocí této komponenty. Pokud nastavíme její vlastnost *Multiline* na *true* můžeme ji využít jako základ poznámkového bloku. Občas potřebujme vybrat položku z určitého seznamu. Tuto možnost nám poskytuje *ComboBox*. K sběru dat nám slouží *DateTimePicker* . Ten uživateli zobrazí kalendář dle námi navolených vlastností. Odpadá tak nutnost kontrolovat, zda je zadané datum validní.

ListView je velmi komplexní ovládací prvek a my si zde představíme jen použité vlastnosti při implantaci této vrstvy. Použili jsme ho při tvorbě seznamů. U některých z nich jsme nastavili jeho vlastnosti zobrazení *view* na *details*. Při této volbě budou zobrazeny názvy sloupce a jednotlivé řádky v nich. Každý řádek představuje objekt *ListViewItem*. Ty jsou uloženy v kolekci *Items*, kterou obsahuje *ListView*. Při nastavení vlastnosti *CheckBox* na *true* nám v řádku přibude položka s políčkem na zaškrtnutí. V *ListViewItem* pak stačí přiřadit vlastnosti *check* hodnotu *true*. Řádek s takto zvolenou vlastností, bude mít políčko zaškrtnuté. V případě přiřazení *false* prázdné.

MainMenu najdeme vždy na spodní straně okna. Muže se skládat s více položek *MenuItem*. Ty se mohou navzájem zanořovat. *MainMenu* je takový rozcestník, který usnadňuje volbu, díky zanořování nám zpřehledňuje výběr položek. *ContextMenu* představuje vyskakovací menu. Vyskočí na uživatele, když na komponentě chvíli přidrží stylu. V našem případě má shodné položky jako standardní menu. Urychluje ovládání aplikace. Při vývoji jsem zjistil, že nelze položky *MenuItem* mít zároveň přidané v *MainMenu* a *CotextMenu*, protože to způsobovalo pád aplikace. Přitom z pohledu logiky návrhu, bylo vše správně.

Komponenta *TabControl* představuje sadu karet, mezi nimiž si můžeme přepínat. Každou kartu lze vytvořit přidáním objektu typu *TabPage* do její kolekce *Controls*. Na *TabPage* lze nanášet další komponenty.

5.1.4.2 Animace pomocí třídy ListView

Při splnění kroku si uživatel odškrtne tuto povinnost ze seznamu. Následně takto označená položka musí ze seznamu zmizet a přesunout se do kategorie splněno. Abychom tuto vlastnost vytvořili, museli jsme umístit na sebe dva seznamy typu *ListView*. Pro lepší orientaci v následujícím odstavci si první seznam označíme *ListView1* a druhý *ListView2*.

Při prvním zobrazení seznamu kroků se zobrazí *ListView*,1 v němž každému kroku je přiřazeno zaškrtávací políčko. Uživatel zaškrtnutím potvrdí, že krok byl vykonán. Program reaguje vyvoláním příslušné funkce. Ta vytvoří *ListView2*, který nechá vykreslit a naopak *ListView1* se neviditelní pro uživatele. Použitý mechanismus vytvoří dojem uživateli v podobě zmizení splněného kroku.

5.1.4.3 Třída ProjektForm

Jedná se o jeden z hlavních pilířů v aplikaci. Při vytváření její instance se vytváří kompletní datová struktura. Jsou načteny data z xml souborů. Z pohledu uživatele na obrazovce cílového zařízení se vykreslí hlavní okno s tlačítky volby (Obrázek 5.3).

Obrázek 5.3 Hlavní okno aplikace

6 Ovládání a instalace

6.1 Instalace

Uživatel nejdříve musí aplikaci nainstalovat. Ta je distribuována pomocí instalačního balíčku .cab souboru. Nalezneme ho na přiloženém CD pod názvem *instalGTD.cab*. Uživatel si balíček zkopíruje do svého mobilního zařízení, například přes ActiveSync. Po té spustí instalaci. Ta vyžaduje minimálně 84kB volného místa k uspěnému naistalování. V nabídce Programy se vytvoří ikonka GTD zástupce ke spuštění.

6.1.1 Požadavky na cílové zařízení

Hlavní požadavek na spravený běh programu je mít na dané platformě naistalováný .NET Compact Framework 3.5. Hardware by měl obsahovat minimálně 300mhz procesor a 64MB dat. Aplikace je koncipována tak, že poběží na pozadí. Rozlišení displeje doporučujeme nejméně 240×320 pixelů.

6.2 Způsoby ovládání

Aplikaci můžeme ovládat pomocí hardwarových tlačítek. Jestliže má zařízení dotykovou obrazovku, lze využít stylus nebo dnes velice oblíbené ovládání pomocí prstů. Nejrychleji se aplikace ovládá pomocí dotyků. Problém nastává při třesu rukou, který může být způsoben, například jízdou v některém z dopravních prostředků. Je poměrně složité se trefit stylusem do určitého ovládacího prvku na obrazovce. Z toho důvodu jsou obě možnosti ovládaní naprosto navzájem zastupitelné. Uživatel si sám muže zvolit, který ze způsobu ovládání je lepší.

6.3 Rychlá volba

Při tvorbě aplikace jsme brali v úvahu nejen komfortní ovládání, ale i čas uživatele spotřebovaný k získání potřebné informaci. Proto se v hlavním okně aplikace nachází nejdůležitější tlačítka (Obrázek 5.3). Během testování jsme zjistili, že uživatel se rozhoduje na základě kontextu a vybírá si kroky z aktuální kategorie. Během tohoto výběru musel uživatel projít čtyřmi nabídkami. Na základě toho jsme přidali do hlavního okna volbu *Možný krok*. Při využití této volby stačí uživateli pouhá dvě stisknutí a dostane se k potřebné informaci. Úspora času je obrovská, oproti původnímu schématu volby.

7 Testovaní

7.1 Funkční testování

Probíhalo již v prvních fázích implementace. Používali jsme emulátory různých verzí prostředí operačního systému. Základní emulátor byl s VGA displejem a operačním systémem 6.1. Pomocí něho jsme odstranili většinu základních logických chyb, jež vznikly při chybné implementaci.

Po dokončení vždy určitého stupně, jsme k otestování zvolili skutečné prostředí. Konkrétně mobilní zařízení Sony Ericsson X1. Mělo nainstalováno operační systém Windows Mobile 6.1 s verzí Frameworku .NET 3.5. Ve většině případu se zde chovala aplikace podobně jako na emulátoru.

Ve zbytku případu nastal problém, který končil pádem aplikace. Původ těchto problému nebylo jednoduché odstranit. I přesto, že jsme se snažili zachytávat chybová hlášení. Compact Framework 3.5 vypíše jen základní zprávu o chybách, kde nás informuje, že podrobnější informace nemůže vypsat, protože se jedná o omezenou verzi Microsoft .NET Framework. To vede k časově náročnější diagnostice chyb v programu.

K odstranění co největšího počtu chyb jsme používali vzorové příklady projektů GTD. Snažili jsme se, ale i o odstranění chyb způsobené nestandardním ovládáním aplikace. Vytvořili jsme si vzorové zátěžové testy. Ty jsme prováděli vždy po určitém rozšíření činnosti aplikace. Zátěžové testy byly neustále aktualizovány, aby byla splněna podmínka nejlepšího možného odstranění chyb. Bohužel nešlo vytvořit jejich automatizaci. Tester musel sám zadávat data dle zvoleného scénáře testu.

7.2 Testování uživateli

Původně jsem předpokládal, že navážeme spolupráci s českou komunitou uživatelů GTD. Nakonec jsem byl rád za tři dobrovolníky. Jelikož jsem za testování neplatil, tak jsem od nich nemohl požadovat přesnou dokumentaci jejich zkušenosti s touto aplikaci. Pokud požadovali rozšíření funkcionality, tak jsem jim ve většině případů vyšel vstříc v následných revizích aplikace. Někteří uživatelé požadovali zvětšení zaškrtávacího políčka. Řešení by bylo vytvořit si vlastní komponentu. Tím bychom, ale snížili kompatibilitu aplikace.

7.3 Akceptační testování

Během zpracování této problematiky mě napadlo moji práci přihlásit do soutěže, abych zjistil, jak si obstojí v konkurenci. Byla to konkrétně soutěž Imagine Cup 2010. Zde se aplikace umístila na druhém místě v celorepublikovém finále. Přínosem pro mě byla konstruktivní kritika od odborné poroty, jejíž složení bylo z řad odborníků informačních technologií. Jejich rady jistě využiji v budoucím zlepšování GTD aplikace. Zaujal jsem porotu především inovační prvky, pojetí tohoto projektu a jeho reálné nasazení v našem každodenním životě. Umístění v soutěži je dobrou reklamou pro tuto aplikaci. Potěšil mě i zájem ze strany veřejnosti.

8 Budoucnost vývoje

Hlavní budoucí cíl vidím ve vytvoření aplikace pro platformu osobních počítačů, případně použít vytvořenou aplikaci. Upravit ji takovým způsobem, aby byla schopna synchronizace s mobilní verzí. Uživatel si daný projekt naplánuje na počítači, kde má komfortnější ovládání. Po té klinutím na jedno tlačítko budou data synchronizována s mobilní aplikaci GTD.

Nyní je uživatelské rozhraní přizpůsobeno ovládání stylusem. To bych rád více uzpůsobil ovládáním pomocí prstů ruky. V další fázi bych se zaměřil na ovládání pomocí lidského hlasu. Všechny tyto změny mají společnou vlastnost zrychlení ovládání aplikace. Uživatel tím uspoří čas, který může využít při plnění projektů. Snaha je minimalizovat čas potřebný k agendě GTD.

Aplikace bez znalosti metody GTD je pro uživatele bezcenná. Uživatel potřebuje nejdříve získat znalosti o této metodě. Bylo by dobré vytvořit e-learningový kurz, kde by si tyto znalosti mohl doplnit. E-learningový kurz bych zaměřil hlavně na praktickou část metody GTD, aby uživatel vytvořil pevné pouto k těmto novým návykům organizace projektů. Jestliže si je nevytvoří, s velkou pravděpodobností se navrátí k původnímu nefunkčnímu systému organizace času.

Firmy se snaží v dnešní době zvýšit efektivnost lidské práce. GTD metoda by pro ně mohla být řešením. Aplikace GTD by se doplnila o profil uživatele. Existoval by hlavní systém. Ten by se synchronizoval s aplikacemi GTD. Vedoucí firmy by tak měl kompletní přehled o všech zaměstnancích. Dozvěděl by se, na jakém projektu daný zaměstnanec pracuje a jaké kroky si naplánoval. Viděl by splněné kroky za dnešní den. Mohl by velice efektivně delegovat jednotlivé projekty. Zaměstnanci by již nemohli odkládat povinnosti na pozdější dobu. Každý by přesně věděl, co má dělat.

K distribuci aplikace bychom mohli použít Windows Marketplace. V katalogu Marketplace bychom ji zařadili do kategorie Produktivita. Byla by to velice efektivní celosvětová propagace. Už jen ten fakt, že aplikaci si může prohlédnout šest miliónů uživatelů a každý z nich může byt potenciální zájemce.

Velké obavy plynou z Windows Phone 7, který ukončuje zpětnou kompatibilitu. Změní se také způsob synchronizace. Bude proto potřeba vytvořit kompletně novou aplikaci.

9 Závěr

První krůčky tvorby této bakalářské práce vedly k nastudování metody GTD. Dále jsem se ji snažil zakomponovat ve svém životě při organizaci času mého dne, abych jí co nejlépe porozuměl. Poté jsem dospěl k názoru, že pokud chceme dosáhnout zlepšení našeho života prostřednictvím metody GTD, musíme chtít změnit náš přístup k životu. Jestliže tuto podmínku splníme a budeme důvěřovat metodě GTD, dosáhneme uvolněného stavu a staneme se pány svého života. Bohužel ne každý je schopen tuto změnu provést.

Aplikace GTD vznikla za účelem maximálně ulehčit správu GTD agendy. Věřím, že se mi toho podařilo dosáhnout a zároveň doufám ve dlouhou životnost této aplikace. Nastínil jsem její budoucnost v komerční sféře. Je zde popsáno možné rozšíření a vylepšení. Aplikace běží na mobilním zařízení, jenž má uživatel neustále při sobě, což je nesporná výhoda.

Mnou vytvořená aplikace zatím nemá moc velkou konkurenci. Sice existují pro Windows Mobile aplikace přizpůsobené pro metodologii GTD, ale uživatelé jsou s nimi nespokojeni, protože nejsou vytvořeny přímo podle specifikace GTD metodologie na rozdíl od mé aplikace.

Během zpracovávání bakalářské práce jsem měl možnost se seznámit s řadou zajímavých nástrojů pro vývoj aplikací. Byl jsem zasvěcen do problematiky mobilního vývoje. Získal jsem podrobné informace o architektuře Microsoft .NET. Mohl jsem si vyzkoušet kompletní řešení vývoje softwarové aplikace od úplného začátku až po konečný produkt.

Literatura

[1] ALLEN, David. Mít vše hotovo: Jak zvládnout práci i život a cítit se při tom dobře. 2001. Brno: Jan Melvil Publishing, 2008. 255 s. ISBN 9788090391284.

[2] Http://www.blackberrycool.com [online]. 2010-02-23 [cit. 2010-05-10]. Gartner Release Breakdown of Mobile OS Market Share. Dostupné z WWW: <http://www.blackberrycool.com/2010/02/23/gartner-release-breakdown-of-mobile-os-market-share/>.

[3] HAMBLEN, Matt.Www.computerworld.com [online]. 2009-09-24 [cit. 2010-05-14]. We screwed up with Windows Mobile.
Dostupné z WWW:
<http://www.computerworld.com/s/article/9138520/Ballmer_We_screwed_up_with_Windows_Mobile_>.

[4] Symbian OS In Wikipedia: the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, , 27. 4. 2010 [cit. 2010-05-01].
Dostupné z WWW: <http://cs.wikipedia.org/wiki/Symbian_OS>.

[5] IPhone OS In Wikipedia: the free encyclopedia [online]. St. Petersburg (Florida): Wikipedia Foundation, 2008-06-24, 2010-05-12 [cit. 2010-05-13]. Dostupné z WWW: <http://en.wikipedia.org/wiki/IPhone_OS>.

[6] Http://developer.android.com [online]. 2010-04-12 [cit. 2010-05-02]. What is Android?. Dostupné z WWW: <http://developer.android.com/guide/topics/ui/index.html>.

[7] Windows Mobile In Wikipedia: the free encyclopedia [online]. St. Petersburg (Florida) Wikipedia Foundation, 26.6 2005, 9.5 2010 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Windows_Mobile>.]

[8] GSM ARENA [online]. 2010 [cit. 2010-05-18]. HTC Touch HD. Dostupné z WWW: <http://www.gsmarena.com/htc_touch_hd-2525.php>.

[9] Socialmediaseo.net [online]. 2010-04-14 [cit. 2010-05-16]. The Most Popular HTC Mobile Phone. Dostupné z WWW:<http://socialmediaseo.net/2010/04/14/htc-touch-the-most-popular-htc-mobile-phone-graph/>.

[10] Adaptable Apps For Windows Mobile. MSDN Magazine [online]. 2008, 6, [cit. 2010-05-09]. Dostupný z WWW: <http://msdn.microsoft.com/en-us/magazine/cc515076.aspx>.

[11] Microsoft [online]. 2007-07-01 [cit. 2010-05-18]. Windows Mobile 6 Professional and Standard Software Development Kits Refresh.
Dostupné z WWW:<http://www.microsoft.com/downloads/details.aspx?FamilyID=06111a3a-a651-4745-88ef-3d48091a390b&DisplayLang=en>.

[12] Submain [online]. 2009-06-15 [cit. 2010-05-13]. GhostDoc. Dostupné z WWW: <http://submain.com/products/ghostdoc.aspx>.

[13] .NET Framework In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida): Wikipedia Foundation, 2003-06-15, 2010-03-03 [cit. 2010-05-18]. Dostupné z WWW:<http://en.wikipedia.org/wiki/.NET_Framework>.

[14] HANÁK, Jan. C# 3.0: Prgoramování na platformě .NET 35. 1. Brno : Zoner Press, 2009. 282 s. ISBN 9788074130465.

[15] RICHTER, Jeffrey. .NET Framework programování aplikací. Praha : GRADA Publishing a.s, 2003. 552 s. ISBN 8024704501.

[16] SHARP, John. Microsoft Visual C# 2008: Krok za krokem. Brno : Computer Press , 2008. 592 s. ISBN 9788025120279, K1596.

[17] HANÁK, Jan . C#:praktické příklady. Praha : GRADA Publishing , 2006. 288 s. ISBN 8024709880.

[18] ESPOSITO, Dino. XML- efektivní programování pro .NET. Praha: Grada, 2004. Serializace XML, s. 596. ISBN 80247077569788024707754.

[19] PETZOLD, Charles . Programování Microsoft Windows Forms v jazyce C#. Brno : Computer Press , 2006. 256 s. ISBN 8025110583.

[20] Http://msdn.microsoft.com [online]. 2010 [cit. 2010-05-10]. Windows Forms. Dostupné z WWW: <http://msdn.microsoft.com/en-us/library/dd30h2yb%28v=VS.80%29.aspx>.

Seznam obrázků

Seznam příloh

Disk přiložený k této bakalářské práci obsahuje:

- Zdrojové kódy aplikace GTD
- Instalační balíček instal-GTD.cab
- ScreenShots aplikace
- Manuál k aplikaci
- Video manuál
- Zdrojový texty této technické zprávy ve formátu pdf a docx